Short Answer:

(1) (4) Is branch prediction really that important? Why or why not?

(2) (4) What is loop unrolling? Is it a valuable optimization technique? Why or why not?

(3) (2)Accurate dynamic branch prediction is more difficult for which type of benchmark, Int or Float?

(4) (3) What is roofline? What is it attempting to measure?

(5) (3) What is the primary difference between Tomasulo’s algorithm and Scoreboarding?

(6) (6) The book lists several things that limit the amount of achievable ILP. List 3 of them, and for each explain why it limits the ILP.

(7) (4) What is a Vector Mask register, and what is it used for?

(8) (8) You have been writing programs for a simple, non-pipelined machine. You have recently received a promotion, and now your job is to write programs for a heavily pipelined, high performance processor. These new programs must execute as fast as possible (the emphasis is on response time, not throughput). Give at least 2 examples of things you might do differently now, and be sure to explain in detail why (what is the problem you are overcoming?)

(9) (8) What is the definition of a basic block? Why is there a desire to create larger ones? Give one example of a way to create a bigger basic block.

(10) (6) The authors list 3 main hardware approaches to supporting multithreading. Briefly describe each one and how it works.

(11) (8) Supporting precise interrupts in machines that allow out of order completion is a challenge. Briefly explain what a precise interrupt is, why it is a challenge in OOO machines, and describe the main technique used today to provide precise interrupts.

(12) (8) Compare and contrast Superscalar and VLIW. Describe each, and list two advantages and one disadvantage of each approach.

(13) (6) Briefly outline how a Vector machine works, and what type of parallelism it is exploiting.

(14) (6) Compilers have trouble optimizing code that involves reads and writes to memory. Why? (The answer has nothing to do with how slow memory is - that is a different problem altogether).

(15) (10) Assume there are 8 logical and 16 physical registers. On the left below is the register mapping upon entering the code sequence. Your job is to fill in the mappings after the execution of the DIVF instruction, including what is on the free list. (Assume that during the execution of this code, no registers are released - in other words, the free list will be shorter at the end than at the beginning.)

BEFORE	
Logical	Physical
0	2
1	4
2	6
3	8
4	10
5	12
6	14
7	0

AFTER	
Logical	Physical
0	
1	
2	
3	
4	
5	
6	
7	

Free Pool: 1,3,5,7,9,11,13,15

Free Pool:

You are given the following code sequence:

```
ADDF   F2,F2,F4
SUBF   F2,F3,F2
MULTF  F4,F3,F2
DIVF   F2,F2,F3
```

Now, rewrite the code sequence below using the actual physical register names instead of the logical ones.

```
ADDF   P__,P__,P__
SUBF   P__,P__,P__
MULTF  P__,P__,P__
DIVF   P__,P__,P__
```

(16) (16 pts) Here is a code sequence.

```
lw      R1, 0(R10)

lw      R2, 4(R10)

add     R3, R2, R1

sw      R3, 20(R0)

sub     R7,R7,1

lw      R4, 4(R10)

lw      R5, 8(R10)

add     R6, R5, R4

sw      R6, 24(R0)

sub     R7,R7,1
```

a) Assuming a standard 5-stage pipeline that does not support hazard detection and does no forwarding, insert as many NOPS as required in order to ensure this code runs correctly. (Remember, writes to the register file occur on the first half of the cycle, and reads occur during the second half).

b) Schedule the code to remove as many stalls as possible. How many cycles does it take now?